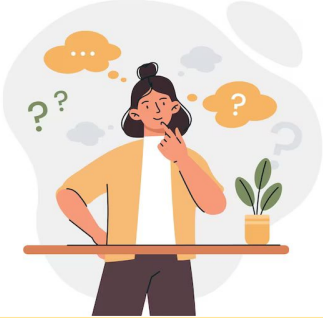
Group 2

Prepared by:
Group 2
Jonathan Chin
Chin Lee Fong
Koh Siew Lee
Man Wai Yee

Agenda

Project Overview



How it works

- Architecture Diagram

Demo Time

- How the app works?
- Logging and Monitoring

Challenges and Conclusion

Our Journey

Dec 2023

Jun 2024

Start of CE5 Class

LOTS OF STUDYING

We are here!

Our Philosophy

We are a large food restaurant company that delivers food to hungry peeps!

To ensure that our customers don't starve, we need to **implement extensive systems' health monitoring to ensure our app is up and running 24/7**. 🔪👩‍🍳

We,✨ the awesome SREs ✨ are here to the rescue to ensure that **systems and application failures are flagged early** and **logs are provided for our engineers to troubleshoot**!

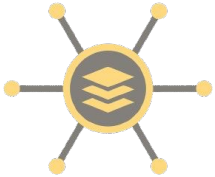
PS: this is a fictious project and for educational purposes only. We're not a real company 🙂

Project Overview

To **monitor a restaurant ordering system application**

(eg health of app, logs generated by app)

To create alarms to **flag any issues with app**



To **automate infrastructure deployment of app**

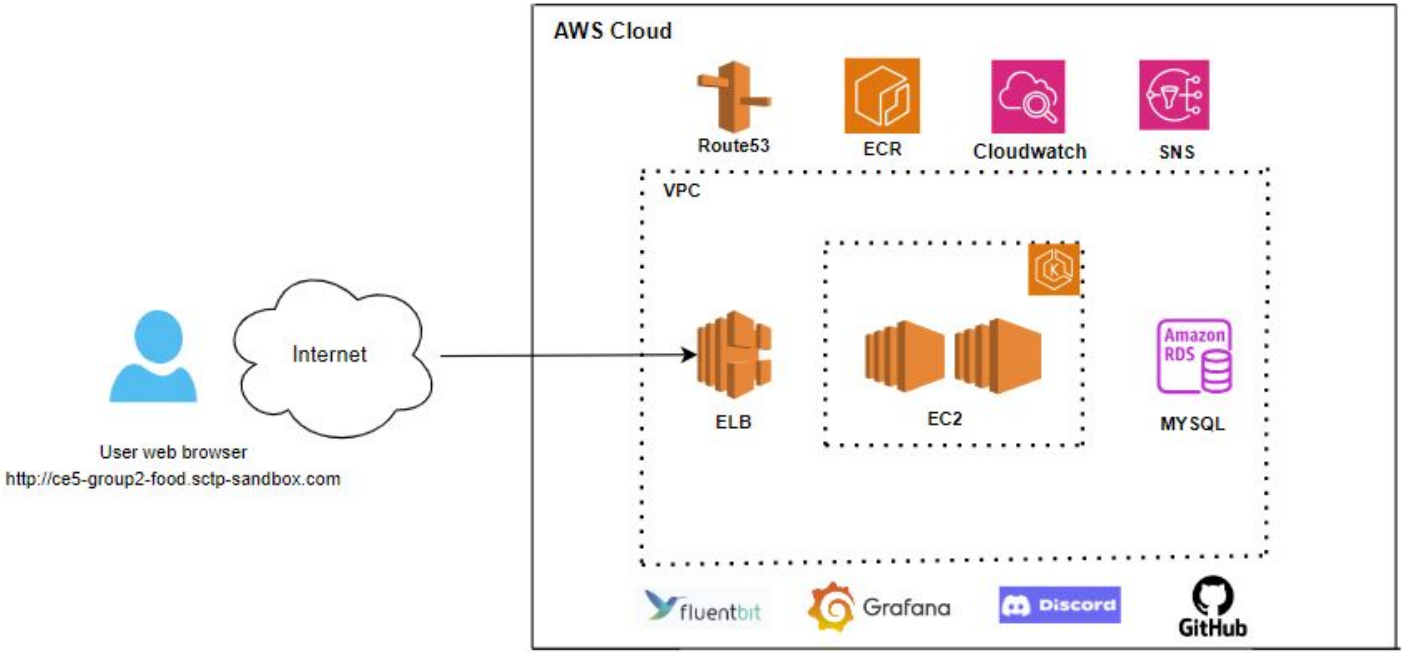
🍳 So let's get things cooking! 🍳

PS: this is a fictious project and for educational purposes only. We're not a real company 🙂

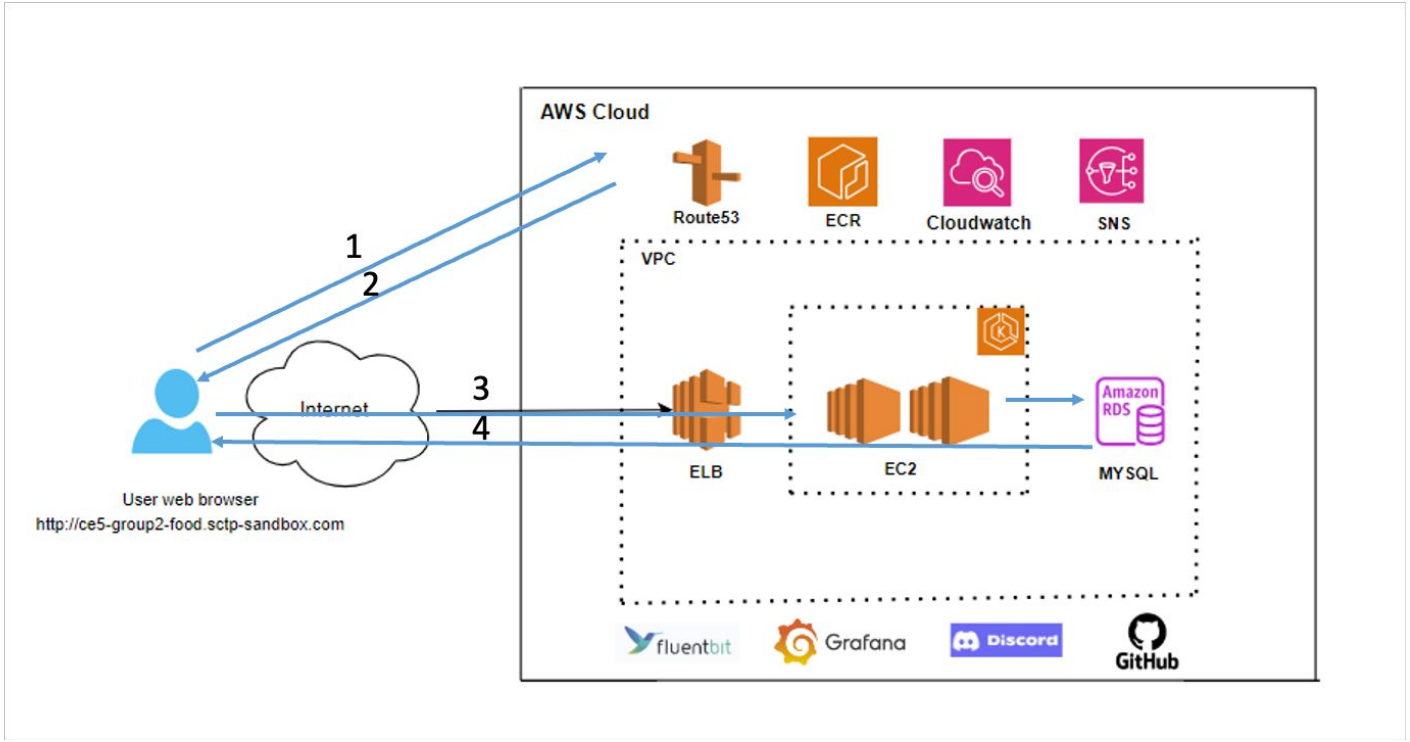
Technology used

- Infrastructure Deployment: Terraform to AWS Cloud
- AWS services used: EKS, RDS, Cloudwatch, Route53, VPC
- Monitoring: Grafana
- Logging: Fluentd
- Alerts: Discord
- CI/CD: Github Actions
    a. Code can push to frontend/backend folder only & will trigger image build to ECR & EKS will auto-apply .yaml file
    b. Code vulnerabilities scanning - frontend, backend, IAC

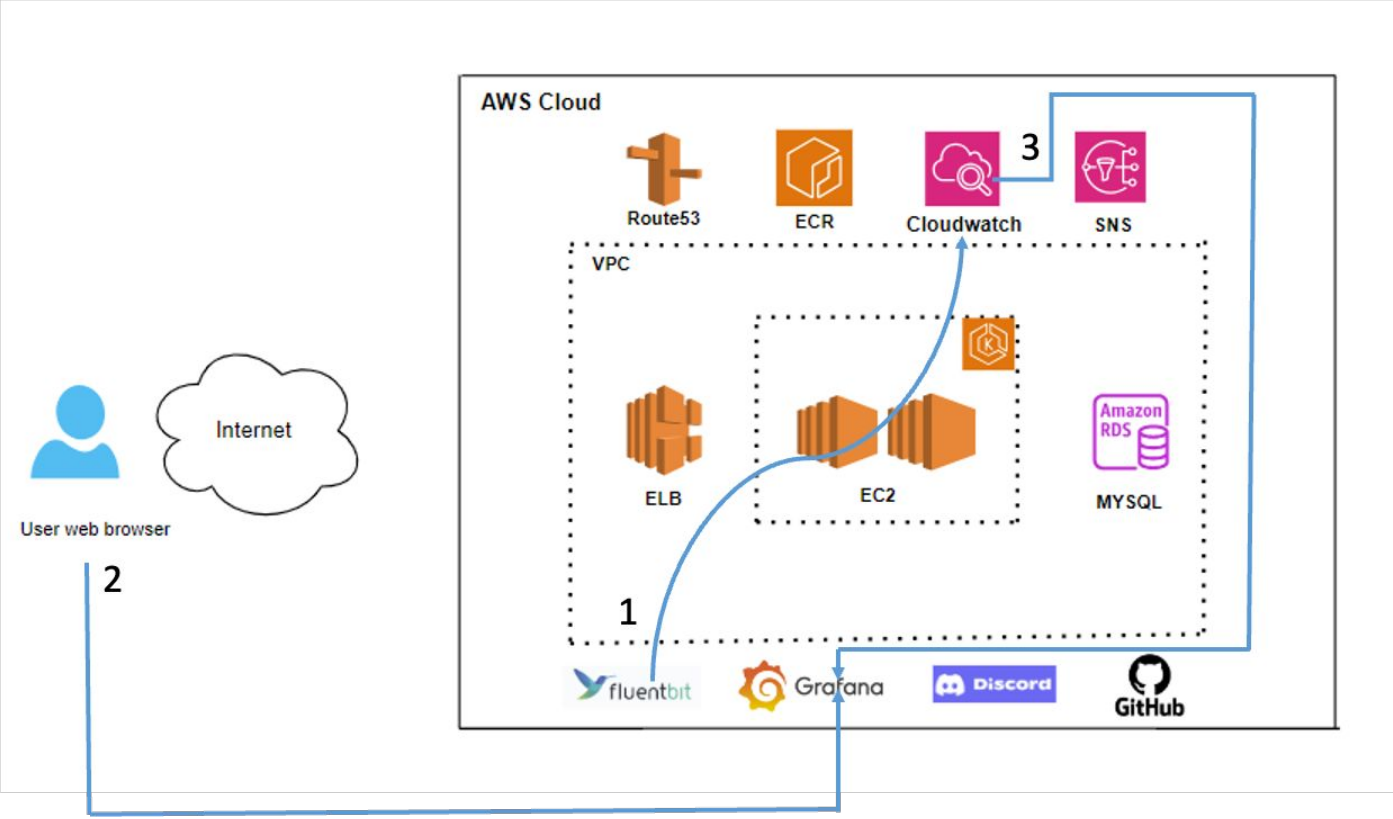
How it works


AWS Cloud
Route53
ECR
Cloudwatch
SNS
VPC
Internet
ELB
EC2
Amazon RDS
MYSQL
User web browser
http://ce5-group2-food.sctp-sandbox.com
fluentbit
Grafana
Discord
GitHub

User Access Application



Logs Collection & Visualization



Demo Time!

Demo Time!

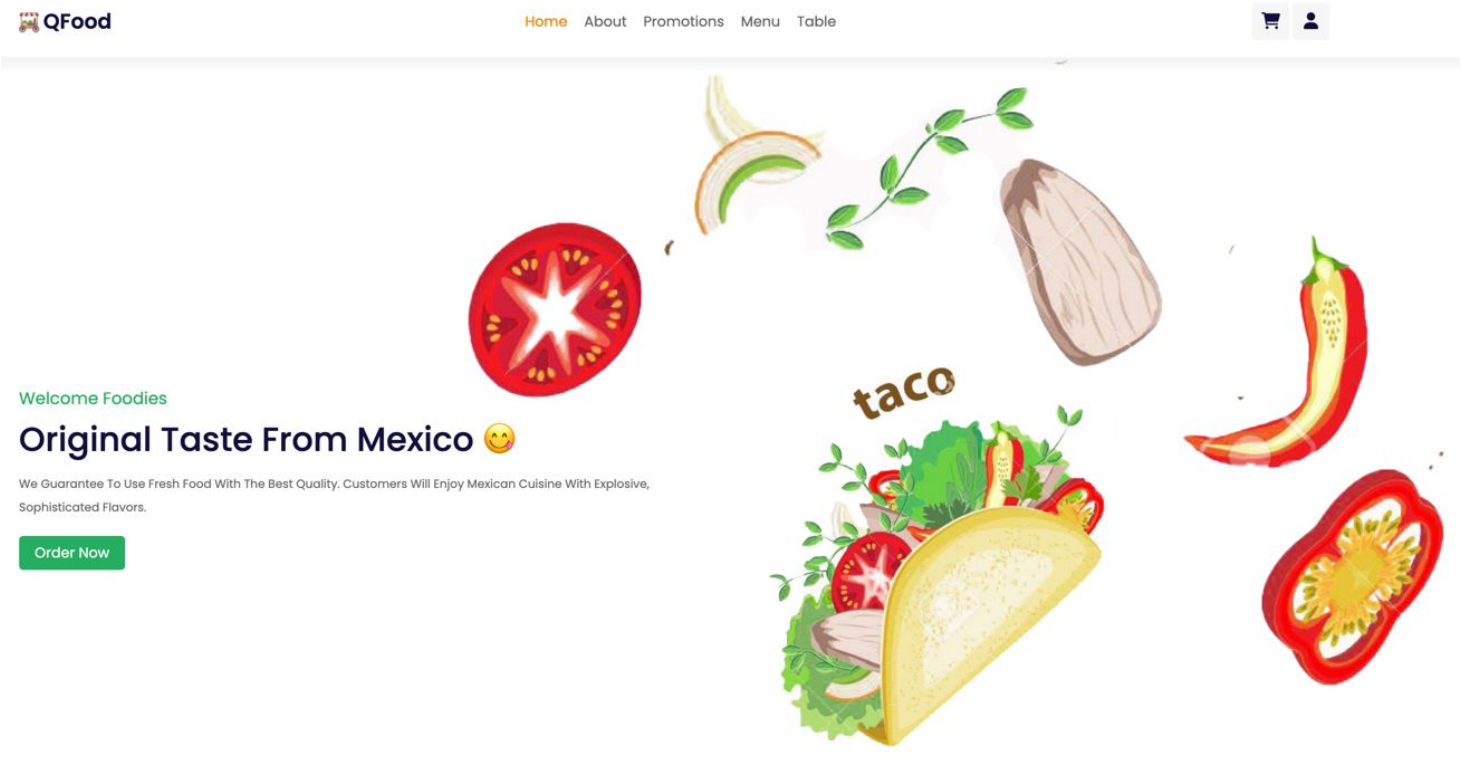
How the app works?

a. Docker images - frontend and backend
b. EKS cluster
c. Application - Create account + Add food
d. Application - backend private RDS (show account is added into SQL)

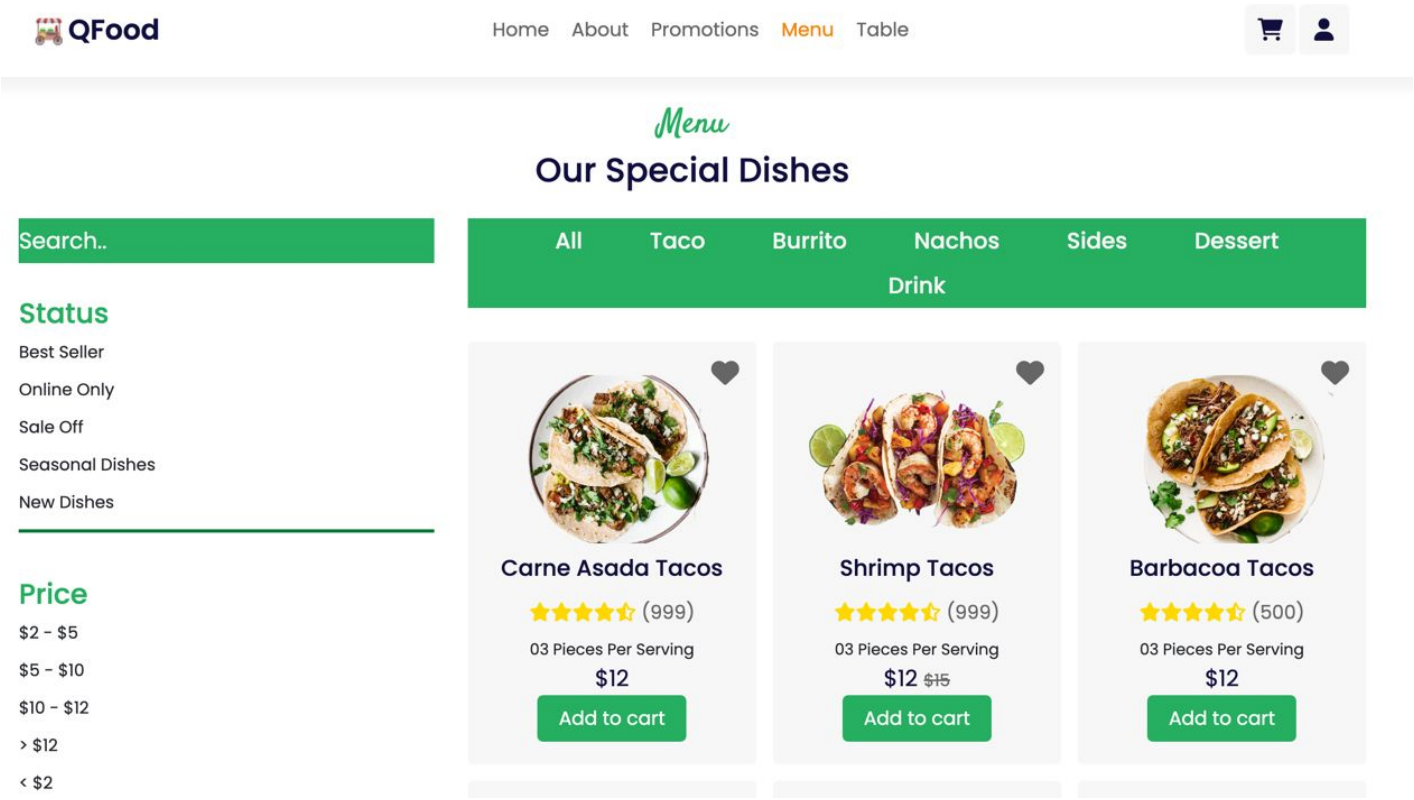
Logging and Monitoring

e. Logging - Grafana dashboard
f. Alarm to team

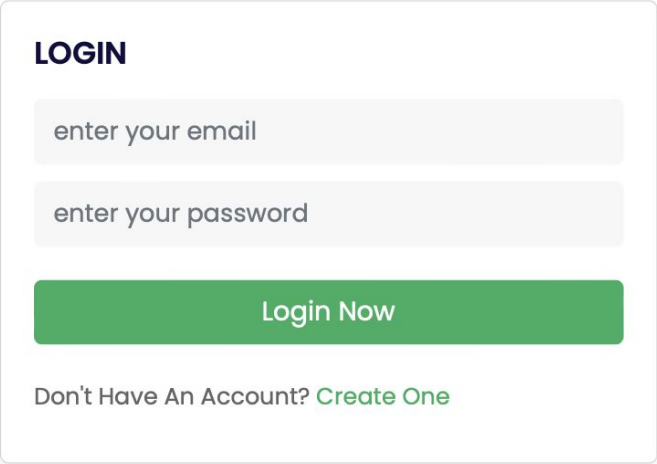
Application - Restaurant Webpage

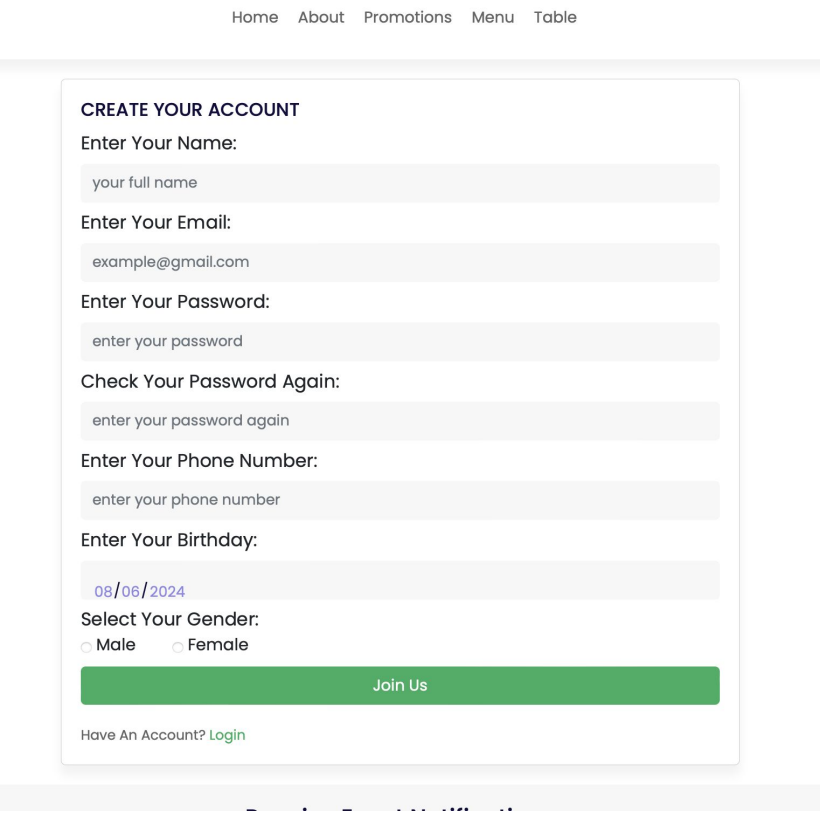


Application - Restaurant Menu



Application - Registering Account



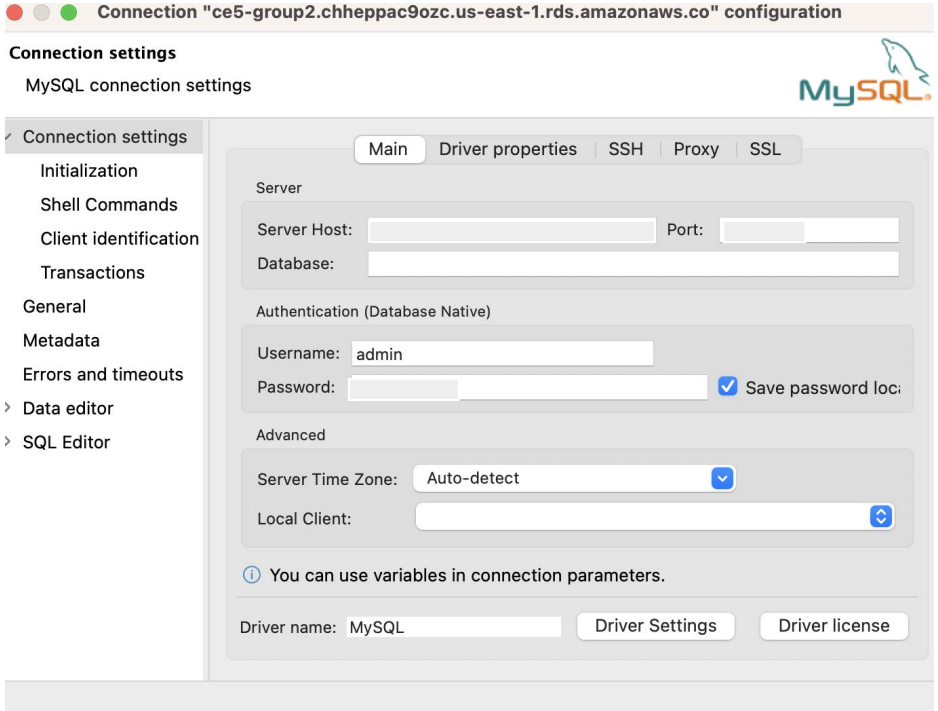


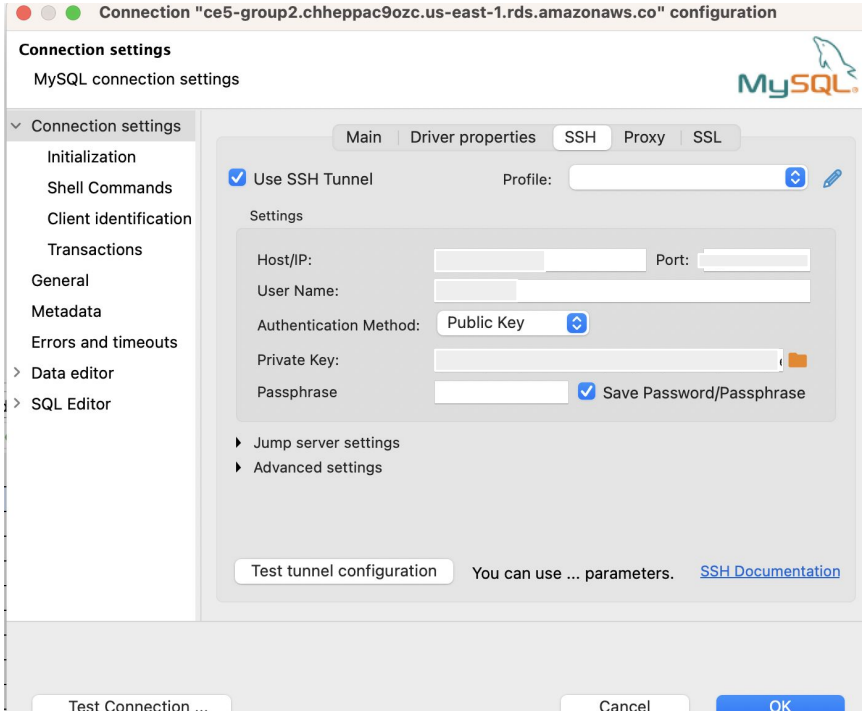
Terraform

- AWS resources deployed through terraform
  - VPC
    - public and private subnets (us-east-1a, us-east-1b, us-east-1c)
    - Internet gateway and NAT gateway
  - MySQL RDS in private subnets
    - Single t3.micro instance
    - With username and password
    - Private subnet groups
    - Security group
  - EKS cluster
    - Private subnets
    - Node group with three t3.medium instances
    - nginx ingress controller (helm)
    - Namespaces for app and monitoring
- Organized into terraform modules so that they are reusable
  - Eg. MySQL can be deployed in the private subnets created in the ‘VPC’ module

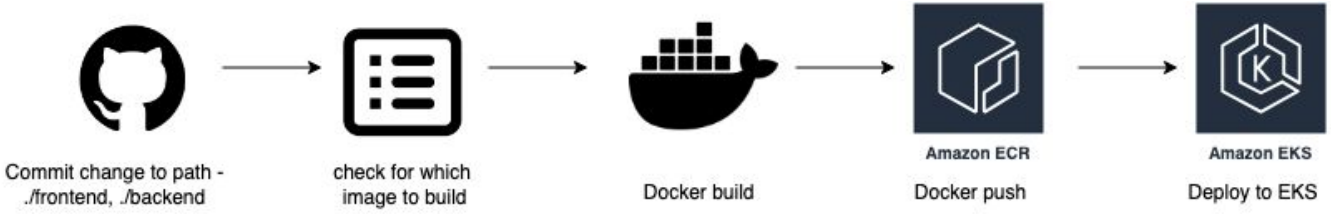
RDS - MySQL

- EC2 bastion host to connect to RDS in private subnet for local connection





CI/CD



1) Build and push docker image to ECR, deploy to EKS
- pipeline is triggered by any github ‘push’ commands to the `frontend` or `backend` folder.
- get changed file(s) and determine the docker image to build based on the path of the changed file(s)
- build docker image and push to ECR
- deploy the EKS yaml file for frontend/backend

2) Code Scanning

- The code_vulnebilities_scan is trigger whenever there's any pull request to the main and develop branch.
- It scans for vulnerabilities in the frontend and backend code, as well as IAC code.

Logging and Monitoring in AWS EKS

Logging

Application Logging

- Winston library for Vue
- Logging level: debug, info, error
- API calls were logged for errors

EKS Logging

- Fluentbit: Published into cloudwatch
- Cloudwatch metrics

Logging - Application Logging

Calling API to get all foods

```
2024-06-08 05:03:35 [info]: Getting all foods...
2024-06-08 05:03:35 [error]: Error in getting all foods: Error: Can't add new command when connection is in closed state
Error: Can't add new command when connection is in closed state
    at Connection._addCommandClosedState (/app/node_modules/mysql2/lib/connection.js:148:17)
    at Connection.query (/app/node_modules/mysql2/lib/connection.js:546:17)
    at getFoods (file:///app/models/FoodModel.js:6:8)
    at showFoods (file:///app/controllers/food.js:15:5)
    at Layer.handle [as handle_request] (/app/node_modules/express/lib/router/layer.js:95:5)
    at next (/app/node_modules/express/lib/router/route.js:144:13)
    at Route.dispatch (/app/node_modules/express/lib/router/route.js:114:3)
    at Layer.handle [as handle_request] (/app/node_modules/express/lib/router/layer.js:95:5)
    at /app/node_modules/express/lib/router/index.js:284:15
    at Function.process_params (/app/node_modules/express/lib/router/index.js:346:12) {
  fatal: true
}
2024-06-08 05:03:37 [info]: Getting all foods...
2024-06-08 05:03:37 [error]: Error in getting all foods: Error: Can't add new command when connection is in closed state
~/Doc/s/ca/g/test/restaurant-ordering-system/de/kubernetes  feature/ci-b..er-image-ecr !2 ?3
```

Getting error message because DB is not up

Logging - EKS Logging

<u>Fluent Bit</u> is a lightweight log processor and forwarder that allows you to **collect data and logs from different sources**, **enrich them with filters** and **send them to multiple destinations** like CloudWatch, Kinesis Data Firehose, Kinesis Data Streams and Amazon OpenSearch Service.

fluentbit

Logging - EKS Logging

Log groups (31)

By default, we only load up to 10000 log groups.

ce5-group2-eks-cluster

4 matches

Exact match

Log group	Log class
/aws/containerinsights/ce5-group2-eks-cluster/application	
/aws/containerinsights/ce5-group2-eks-cluster/dataplane	Standard
/aws/containerinsights/ce5-group2-eks-cluster/host	Standard
/aws/eks/ce5-group2-eks-cluster/cluster	

*Application and pod logs (mainly used)

Amazon EKS **control plane logging** provides **audit and diagnostic logs** directly from the Amazon EKS control plane to CloudWatch Logs in your account.

Logging - EKS Logging

Log streams (100+)

By default, we only load the most recent log streams. Load more.

Delete | Create log stream | Search all log streams

Filter loaded log streams or try prefix search

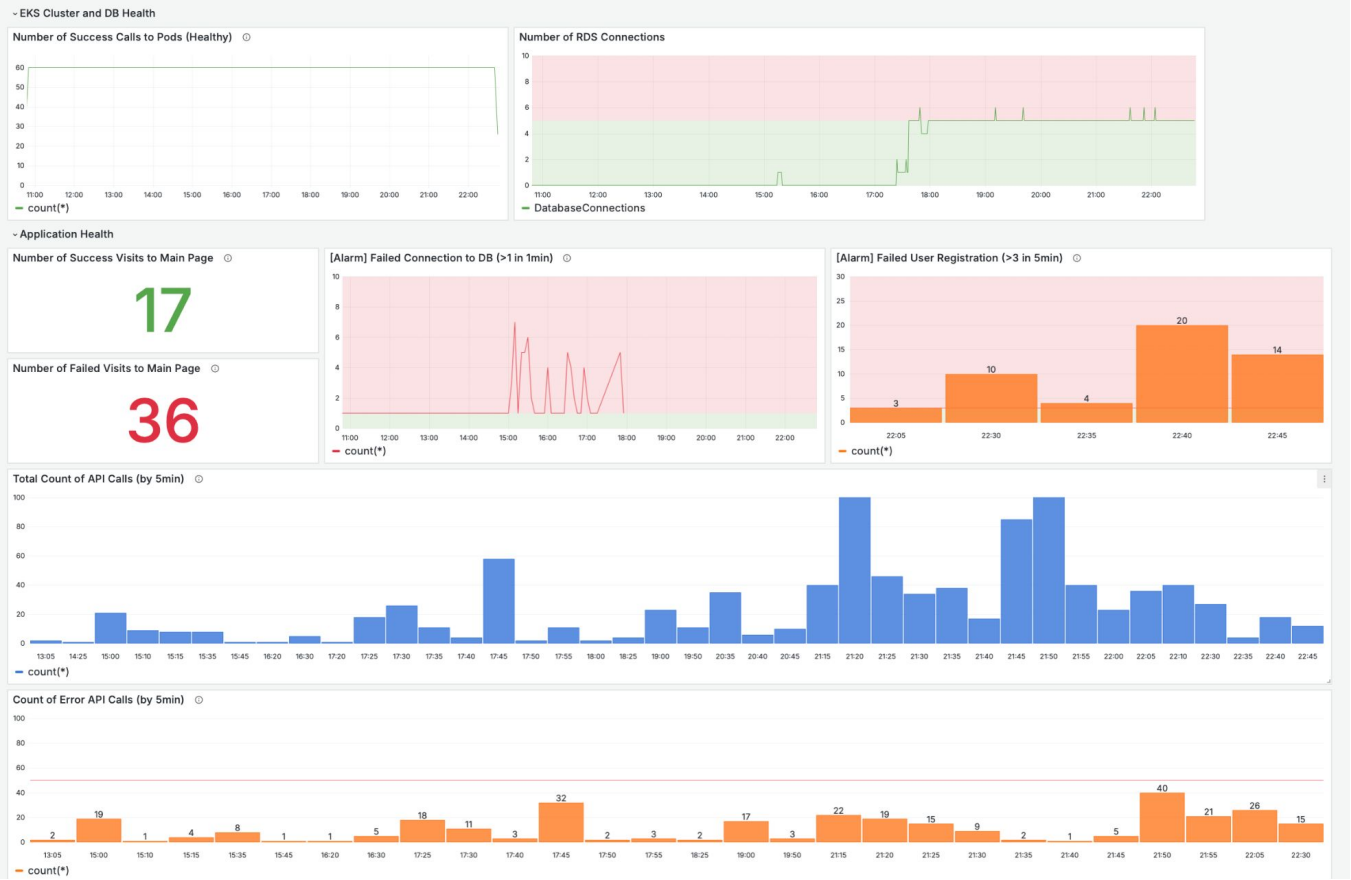
Exact match | Show expired | Info

1 2 ...

Log stream	Last event time
ip-10-0-3-127.ec2.internal-application.var.log.containers.restaurant-app-74bd4c4c58-l74lt_restaurant_frontend-container-0b01dfb248f5f4897f1d86ff2a2a0b12bcfe1	2024-06-13 22:52:43 (UTC+08:00)
ip-10-0-1-37.ec2.internal-application.var.log.containers.restaurant-api-7db8d87888-wdbvp_restaurant_backend-container-855663d469cba5c5bdf121f2bc7294ddeb03	2024-06-13 22:51:17 (UTC+08:00)
ip-10-0-3-127.ec2.internal-application.var.log.containers.grafana-7b5f56f99c-5lzm4_monitoring_grafana-45c15311355bd1b837418a965c88b19738a02f5a62eadf8a4	2024-06-13 22:47:11 (UTC+08:00)
ip-10-0-3-127.ec2.internal-application.var.log.containers.nginx-ingress-controller-default-backend-79b9766c56-bx2ht_default_default-backend-b0a890e144b98461a0	2024-06-13 22:38:48 (UTC+08:00)
ip-10-0-3-127.ec2.internal-application.var.log.containers.nginx-ingress-controller-5857568d85-8krgv_default_controller-c8bc4d68f7b5b087fc0043fb55b0653af1973d	2024-06-13 22:35:04 (UTC+08:00)
ip-10-0-1-37.ec2.internal-application.var.log.containers.restaurant-api-7db8d87888-5trd9_restaurant_backend-container-8b2c34f531efad5f6bf061aceac912d926227b	2024-06-13 22:14:15 (UTC+08:00)

Logstreams from frontend, backend, ingress-controller, ingress-controller-backend pods

Monitoring - Grafana Dashboard



Alarms that will be flagged

Alarms	Reasons	Conditions
Error in DB Connection	Customers won't be able to choose their food, login with existing account	db-connection-error > 1 for 1 datapoints within 5 minutes
Error in user registration	Customers need to register for an account before buying food	error in num-of-failed-user-registration > 3 for 1 datapoints within 5 minutes
Error in accessing mainpage	Customers won't be able to order food, which will lead to unhappy hungry customers	num-of-failed-visitors-to-main-page > 5 for 1 datapoints within 5 minutes

Amazon CloudWatch

Amazon SNS

Amazon Lambda

Notification Channels

ALARM: "Error accessing main page" in US East (N. Virginia) Inbox ×

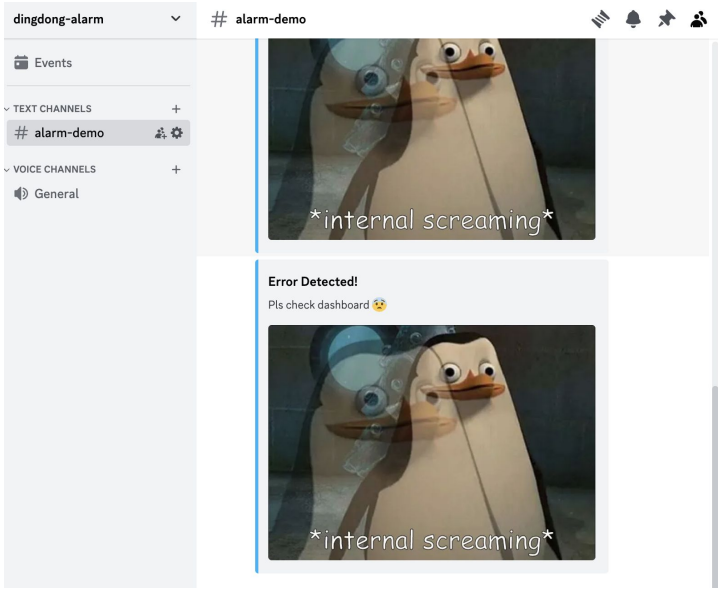
AWS Notifications <no-reply@sns.amazonaws.com>
to me

You are receiving this email because your Amazon CloudWatch Alarm "Error accessing main page" in the US East (N. Virginia) region has datapoints [6.0 (10/06/24 13:58:00)] was greater than the threshold (5.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Monday 1

View this alarm in the AWS Management Console:
https://us-east-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=us-east-1#alarmsV2:alarm/Error%20accessing%20main%20pa

Alarm Details:

Email



Discord Channel

After debugging…

```
39
40    // create user
41    export const createAccount=(req,res)=>{
42        logger.info("Creating a new user account...");
43        const data = req.body;
44
45        const userEmail = JSON.stringify(data, ["user_email"]);
46        logger.info(`User email: ${userEmail}...`);
47
48
49        if (userEmail === '{"user_email":"abcd@gmail.com"}' ) {
50            const errorMessage = "Cannot register user, please try again with a different email address.";
51            logger.error(errorMessage);
52            return res.status(400).json({ error: errorMessage });
53        }
54
```

Challenges

kubernetes

Complexity of k8s

Kubernetes is a **complex container orchestration system** and has a **steep learning curve.** Eg Understanding architecture, components (like pods, services, deployments, namespaces), and operational principles.

Frontend Backend

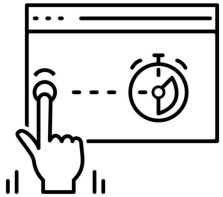
Frontend and backend app were **not production-ready** and **required more env variables** to work correctly. It required **additional configuration and setup to be deployed in k8s.**

aws

AWS Implementations

EKS **integrates with other AWS services**, eg IAM for authentication, ELB for load balancing, and CloudWatch for logging and monitoring. **Understanding how these services interconnect adds another layer of complexity.**

Moving Forward



UNIT TESTING

Unit/integration tests

Add unit/integration tests for the frontend and backend application as part of the CI/CD pipeline.

AWS IAM

Security Measures

Implement **more stringent security measures** *(eg using IAM roles for github actions, grafana dashboard access, and EKS cluster access.)*

Authentication + Authorization

Implement **robust authentication** *(e.g., OAuth2, JWT)* and **authorization** *(role-based access control)* mechanisms for the **backend API calls.**

Moving Forward

Sensitive Information

Store sensitive information *(API keys, database credentials)* securely. Using AWS Secrets Manager or AWS Systems Manager Parameter Store.

Database Migrations

Automate the migrations for MySQL for any schema changes.

Materials

Github repo: https://github.com/StrawberriCake/restaurant-ordering-system

🌯🥗🌯🌯🥗🌯🌯🥗🌯🥗🌯🌯🥗🌯🌯🥗🌯🥗🌯🌯🥗🌯

Thank You! :)

🌯🥗🌯🌯🥗🌯🌯🥗🌯🥗🌯🌯🥗🌯🌯🥗🌯🥗🌯🌯🥗🌯

🍳

Pls support our restaurant:
http://ce5-group2-food.sctp-sandbox.com

PS: this is a fictious project and for educational purposes only. We're not a real company 🙂